(2)

USAISEC

US Army Information Systems Engineering Command
Fort Huachuca, AZ 85613-5300

U.S. ARMY INSTITUTE FOR RESEARCH
IN MANAGEMENT INFORMATION,
COMMUNICATIONS, AND COMPUTER SCIENCES

A Distributed TDMA Rescheduling Procedure for Mobile Packet Radio Networks (ASQB-GI-91-008)

JANUARY 1991

AIRMICS
115 O'Keefe Building
Georgia Institute of Technology
Atlanta, GA 30332-0800

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188
Exp. Date: Jun 30, 1986

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS NONE
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT N/A
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ASQB-GI-91-008	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION AIRMICS 6b. OFFICE SYMBOL (*If applicable*) ASQB-GI	7a. NAME OF MONITORING ORGANIZATION N/A
6c. ADDRESS (City, State, and Zip Code)	7b. ADDRESS (*City, State, and ZIP Code*) N/A
8b. NAME OF FUNDING/SPONSORING ORGANIZATION AIRMICS 8b. OFFICE SYMBOL (*If applicable*) ASQB-GI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*) 115 O'Keefe Bldg. Georgia Institute of Technology Atlanta, GA 30332-0800	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (*Include Security Classification*)
A Distributed TDMA Rescheduling Procedure for Mobile Packet Radio Networks (UNCLASSIFIED)

12. PERSONAL AUTHOR(S)
Major(P) David S. Stevens and Mostafa H. Ammar

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Technical Paper	FROM_____ TO_____	January 28, 1991	28

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUBGROUP	Time Division Multiple Access, TDMA, Packet Radio Networks, Channel Access Protocol

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

This paper presents a solution to the problem of simultaneously allowing a mobile node to locally determine a new TDMA collision free transmission schedule while at the same time maximizing the utilization of the available bandwidth to minimize delay. Previous works were able to do only one or the other.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL MAJ(P) David S. Stevens	22b. TELEPHONE (*Include Area Code*) (404) 894-3110 22c. OFFICE SYMBOL ASQB-GI

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

This research was performed by Major(P) David S. Stevens currently assigned to the Army Institute for Research in Management Information, Communications and Computer Sciences (AIRMICS) and by Dr. Mostafa H. Ammar a professor at the College of Computing located at the Georgia Institute of Technology. Dr. Ammar is supported by NSF grant NCR-6804850. The presentation of this paper at the 1991 IEEE International Conference on Communications was made possible by the support of AIRMICS, the RDTE organization of the U.S. Army Information Systems Engineering Command (USAISEC).

THIS REPORT HAS BEEN REVIEWED AND IS APPROVED

s/ _____

Glenn Racine, Chief
Computer and Information
Systems Division

s/ _____

John R. Mitchell
Director
AIRMICS

A Distributed TDMA Rescheduling Procedure for Mobile Packet Radio Networks

David S. Stevens and *Mostafa H. Ammar*

January 28, 1991

Abstract

Packet radio networks provide two features not present in a wire-based network - mobility and a broadcast channel. Time Division Multiple Access (TDMA) protocols provide packet radio networks with two features that facilitate efficient communications. First, they eliminate the possibility of collisions on the broadcast channel. Second, they allow for the spatial reuse of the radio channel bandwidth by permitting more than one node to transmit at once. However, the goal of maximizing the use of the bandwidth seems to conflict with the goal of allowing mobile nodes to *locally* reallocate themselves TDMA slots so that collisions will not occur. We present a procedure that permits a node to move and then reallocate itself a transmission slot without involving the entire network. In our procedures the channel over which control packets are exchanged is shared and unreliable. Therefore the resulting TDMA schedule may not be coliision free. We present a collision resolution algorithm to correct these problems. Finally a procedure by which nodes can allocate themselves additional transmission slots, if they are available and which maximizes bandwidth utilization, is given.

M. H. Ammar is supported by NSF grant NCR-8604850 and Major D. S. Stevens is supported by the U.S. Army Institute for Research in Management Information, Communication and Computer Sciences.

1 Introduction

Packet radio networks have unique features that wire-based networks lack, that being mobility and a broadcast medium. Time Division Multiple Access (TDMA) protocols provide packet radio networks with collision free communications and permit spatial reuse of the radio channel by allowing more than one node to transmit at once. Many algorithms exist for allocating transmission rights to nodes in a packet radio network that produce a TDMA collision free schedule [1, 2, 3, 4, 5, 6, 7]. The different methodologies can be grouped into two general strategies for assigning transmission rights to nodes - *Node* and *Link* allocation. Node allocation assigns slots to a node during which it can broadcast packets, collision-free, to any one or group of its neighbors. In a link allocation strategy, unique time slots are allocated to a node for each directed link it has to a neighbor. A node can transmit to a neighbor only during the time slot assigned to the directed link for that neighbor.

When a TDMA protocol is coupled with mobile nodes, established collision free schedules deteriorate. To accommodate mobile nodes Chlamtac and Pinter in [8] proposed a distributed algorithm that reestablishes a collision free environment for node allocation schemes. In [9] a solution is presented for link allocation scheduling while the work described in [6] offers a solution for both node and link allocation protocols. However, these distributed algorithms require long TDMA frames and allocate only a single slot per node or directed link per TDMA frame, and thus may result in a schedule that does not fully use the available broadcast channel. In static radio networks, centralized algorithms with global topological information can make use of elaborate heuristics, such as finding a maximal clique, to allocate more than one slot to a node or directed link per TDMA frame [1, 2, 7, 10]. (Finding an optimal schedule has been shown to be an NP-complete problem [5, 11].) In [11], Ephremides and Truong, developed a distributed solution to maximize the bandwidth utilization that uses a frame length equal to the number of nodes in the network and a priority scheme based on a node's ID. Still, as they point out, a shortcoming of their solution is that it does not address changing connectivity except by repeating the algorithm from scratch. In addition, the priority scheme used can result in an unfair allocation of time slots with a single node controlling over 80% of the bandwidth within its *broadcast zone* (defined as the set of nodes that are one and two hops away [11]). Chlamtac and Lerner [12] developed a distributed "maximally fair" allocation algorithm for link allocation that relied on the maintenance of a minimal spanning tree and the continuous calculation of a saturation measure. Their solution insured global fairness at the expense of not using all slots and when a node moved their fairness criteria could cause all nodes to become involved in the reallocation process.

It appears that algorithms that address the maximal use of the bandwidth do not adequately handle topological changes. (By *maximal* we mean that an additional slot cannot be allocated to a node for transmission without causing collisions.) Those that can locally adapt their schedules to accommodate mobile nodes do not use all the available bandwidth. In this paper we describe a procedure for adjusting TDMA slot allocation to nodes in a mo-

bile packet radio network where maximal use of the bandwidth is achieved, rearrangement of the TDMA schedule involves only a few nodes, and where control messages are transmitted over a shared error-prone channel. The procedure is an *optimistic* one in that nodes make a best effort at assigning themselves collision free slots. Special recovery procedures are defined in case inconsistencies causing collisions are discovered in the slot assignments.

The paper is organized as follows: Section 2 describes the packet radio network model underlying our investigation. In Section 3 we describe the problems that a mobile node or a new node entering the network can create for an established collision free TDMA schedule. In Section 4 we outline and motivate our approach to solving the problem. Section 5 describes the procedure that is invoked when a node changes location. In Section 6 we discuss the recovery procedures invoked when a node discovers inconsistencies in the TDMA slot assignments. In Section 7 we present a distributed procedure that can be used by nodes to allocate slots to achieve maximal fair allocation of the TDMA frame. Section 8 contains a discussion of the parameters affecting the performance of the procedures. Some concluding remarks are given in Section 9.

2 Network Model

The radio network is modeled by an undirected graph $G(V, E)$. The vertices $V = \{v_1, v_2, ..v_n\}$ represent the nodes of the network and are the individual packet radios, where $\mid V \mid = N$, the number of nodes in the graph. Each undirected edge $e \in E$ is interpreted as two antiparallel directed edges. Thus an edge between u and v implies that u can receive every signal transmitted by v, and v can receive every signal transmitted by u.

We will assume that packets are of a constant length, and we will partition time into constant size time slots that are equal to the packet length plus the maximum propagation delay. These slots are organized into TDMA frames where each node in the network is allocated at least one slot per frame. One methodology for allocating time slots in a packet radio network involves traversing the network using a depth first strategy and using a greedy selection algorithm to assign a single time slot to each node. A time slot is available if assigning it does not result in a collision. To avoid collisions, the following two conditions must hold:

C1. Node v does not transmit in the same period during which a neighbor is transmitting to it. (A node cannot send and receive simultaneously.)

C2. Only one neighbor of node v can transmit to it during any one period of time. (A node cannot simultaneously receive two transmissions.)

A slot assignment is *consistent* if it meets the above conditions and it is *inconsistent* otherwise. We assume that the network is initialized with a consistent slot assignment that

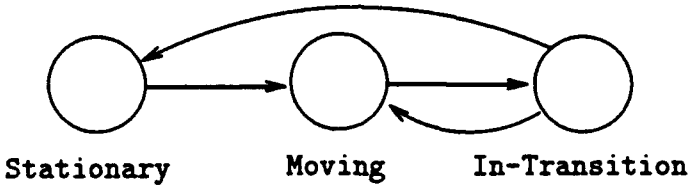

Figure 1: Possible State Transitions

allocates at least one slot per node per TDMA frame. Any of the node allocation schemes mentioned in Section 1 may be used for this purpose.

We assume that nodes periodically broadcast *status packets* to their neighbors. These packets will contain information about the slot assignments for this node, as well as its neighbors. Through these status packets each node is made aware of the status of all nodes within its broadcast zone. We also assume that all packets contain source and destination addresses.

All network nodes are mobile. However, we assume that they do not move continuously. In particular, we assume that the time the node remains in one location is long compared to the time required for it to rearrange the TDMA schedule once it has completed the move. A node is said to be in either of three states:

- **Stationary:** if it is not moving and is assigned at least one slot per frame in which it can transmit.
- **Moving:** if the node is moving.
- **In-Transition:** if the node is not moving, is not assigned any slots, and is in the process of determining a slot assignment.

The possible transitions between these states are illustrated in Figure 1.

3 The Problem

To understand the problems that mobile nodes can cause, consider the small network in Figure 2. Slot assignments were made using a distributed greedy selection node allocation algorithm [13], which assigns a single slot per node per frame resulting in a TDMA frame that is six slots long. Note that some nodes can transmit in additional slots without collisions resulting. For example, node A also can be assigned slot 6 and node D can be assigned

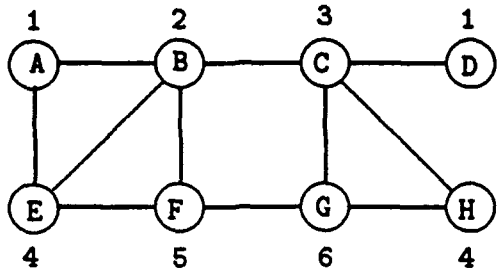

Figure 2: Example Network Using Node Allocation

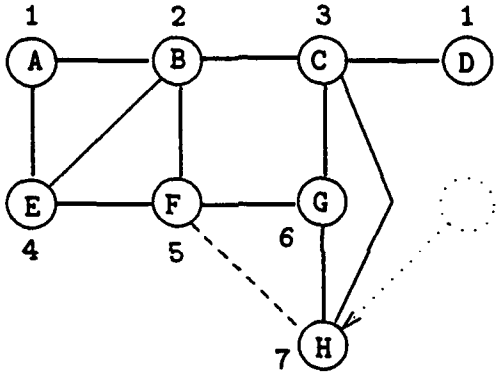

Figure 3: Example Network with Mobile Node

slot 5. However, for the time being we assume that these additional assignments are not made.

Now assume that node H relocates and establishes an additional communication link with node F (See Figure 3). Node H must reassign itself a new slot to reestablish a collision free schedule. Yet in the current TDMA frame, a free slot does not exist. (A new node entering the network in this situation would experience a similar problem.) The only solution is to extend the frame length either by rerunning the initial allocation algorithm or by coordinating a switch to a longer frame length. These solutions, while providing node H with a time slot, unfortunately require the entire network's involvement. In addition, should node H return to its original location, the question becomes what do we do with the unneeded time slot 7? Should the frame length be reduced or should node H be permitted to continue using it? Constant altering of the frame length is disruptive to the network. Yet, not doing so makes the frame length longer than necessary.

In [13] it was shown that longer frame lengths, even if all slots are allocated, usually result in poorer delay performance when each node is assigned exactly one slot per frame. Thus if the network uses a frame length longer than required, it becomes necessary to allocate more than one slot per node whenever possible to offset the reduced performance that will be experienced. For instance, in the example when node H returns to its original location it can be permitted to continue using slot 7 and its original slot 4. To improve performance further, node A can be allocated slot 6, node D can be assigned slot 5, and node B can now also be assigned slot 7. While these optimizations maximize the use of the bandwidth, it leaves the network vulnerable to the original problem of time slots not being available for mobile nodes or for a new node entering the network.

4 Solution Approach

4.1 Requirements

Our approach to handling TDMA schedule rearrangement in a mobile environment attempts to satisfy the following set of requirements:

1. As small a number of nodes as possible should be involved in the assignment of a new slot to mobile nodes.
2. The scheme should make maximal use of the TDMA slots.
3. The transmission of required control messages, although allocated separate bandwidth, cannot be assumed reliable.

We first observe that in rearranging the TDMA schedule as a result of the network's reconfiguration, we may require a change in the TDMA frame length (see Section 3). A change of frame length needs to be communicated to the entire network to maintain proper synchronization. Therefore, the first requirement above dictates that the TDMA frame length must remain constant despite network reconfiguration. To achieve this we call upon a result derived in [8] where it is shown that the maximum TDMA frame length required for any network is given by:

$$L^* = min\{|N|, l_{max}^2 + 1\} \quad (1)$$

where N is the maximum number of nodes in the network and l_{max} is the maximum number of neighbors that a node can have. Typically, when allocating a single slot per node, the frame length required can be significantly less than L^* (see [13]). However, we use a TDMA frame with *at least* L^* slots to guarantee that a consistent assignment (satisfying C1 and C2) can always be found without having to increase the frame length regardless of the network topology.

Slot	A	B	C	D	E	F	G	H
1	P	-	-	P	-	-	-	-
2	-	P	-	-	-	-	-	-
3	-	-	P	-	-	-	-	-
4	-	-	-	-	P	-	-	P
5	-	-	-	S	-	P	-	-
6	S	-	-	-	-	-	P	-
7	-	S	-	-	-	-	-	-
8	-	-	S	-	-	-	-	-

Table 1: Possible Node Allocation Schedule

If we use a frame length of L^* or more and only allocate a single slot per node, the second requirement above will not be met, i.e., some nodes can be allocated additional slots in which they can undertake collision free transmissions. In our approach each node may have *multiple* slots assigned to it per frame. In this assignment, each node is allocated exactly one *primary* slot per frame and zero or more *secondary* slots. For example Table 1 shows one possible assignment of slots in the network of Figure 2. Each node is allocated exactly one primary slot (indicated by the letter P). Nodes A, B, C and D are each allocated one secondary slot (the letter S). The notation "-" indicates that the slot is not available to a node because transmitting in it would result in a collision. Note that this assignment is maximal as no more slots can be assigned while still satisfying conditions C1 and C2.

Finally, we observe that any rearrangement of the TDMA slot allocation will require the use of control messages. When a node moves there is now no guarantee that it can communicate without collisions. To overcome this problem, some solutions assume that control packets will be correctly received within a finite amount of time or do not collide [8, 9, 14]. Other solutions have avoided the problem by using a frame length equal to the number of nodes [11], which always insures a unique collision free slot for each node, or a control channel divided equally among the network members [3], which guarantees that control messages will never collide. Our solution to the problem does not make these assumptions, nor do we require that all control packets be received correctly.

We assume that control messages are allocated a separate channel in the form of an extra slot in each TDMA frame. Use of this channel implies that control messages will not collide with data packets, yet they may collide with other control messages. Furthermore, we make no assumptions about the reliability of this control channel. Control packets are not guaranteed to reach their destinations even if no collisions occur. We require that our procedures function correctly under these adverse conditions.

The implication of this can be made clear by considering a node that moves to a new location. In order to assign itself new transmission slots, it needs to learn about the slots assigned to nodes in its broadcast zone. This has to be done via the exchange of control messages. The unreliability of this exchange implies that it is impossible for a node to be 100% certain that it has acquired complete knowledge of its broadcast zone in any finite amount of time. This situation is exacerbated by the fact that in a mobile network any information that a node possesses about its broadcast zone may be incorrect when that information is used. Therefore, we only require that when a node moves to a new neighborhood, it exerts its best effort to gather information about its broadcast zone and to insure a consistent TDMA schedule. Inconsistencies in the TDMA schedule (which will cause collisions) are detected and recovered from by ongoing processes in all nodes.

4.2 Components

To meet the requirements set forth in the above discussion we define three procedures to be implemented in each node:

1. *Primary_Slot_Assignment (PSA):* Used to assign a mobile node a new primary slot after it has located to a new neighborhood. Recall that we defined a Stationary node as one that is in possession of a primary slot allocation.

2. *Collision_Resolution (CR):* Used to recover from violations of conditions C1 and C2 whenever they are detected in a Stationary node.

3. *Secondary_Slot_Assignment (SSA):* Used to assign secondary slots whenever a node, which is Stationary and is not experiencing collisions, detects that such slots are available.

The PSA procedure is run according to the flow chart shown in Figure 4. Whenever a node starts moving it stops transmission in all slots. The PSA procedure is run when the node has relocated to a new position. The node is thus unable to send data packets while it is moving and, as we will see later, also for a short time after it has completed moving. We thus implicitly assume that such time is short compared to the time between consecutive moves by the same node.

The SSA and CR procedures are run according to the flow chart in Figure 5. Note that none of the three procedures can be running simultaneously. Each node maintains two boolean variables: *Collision_Detected*, and *Secondary_Available*, which are set to True if the conditions indicated are met (more on this later). These conditions are checked periodically once every X frames, where X should be a random integer chosen uniformly between some minimum and maximum number. This will help prevent synchronous running of these procedures among different nodes and will reduce the likelihood of inconsistent TDMA

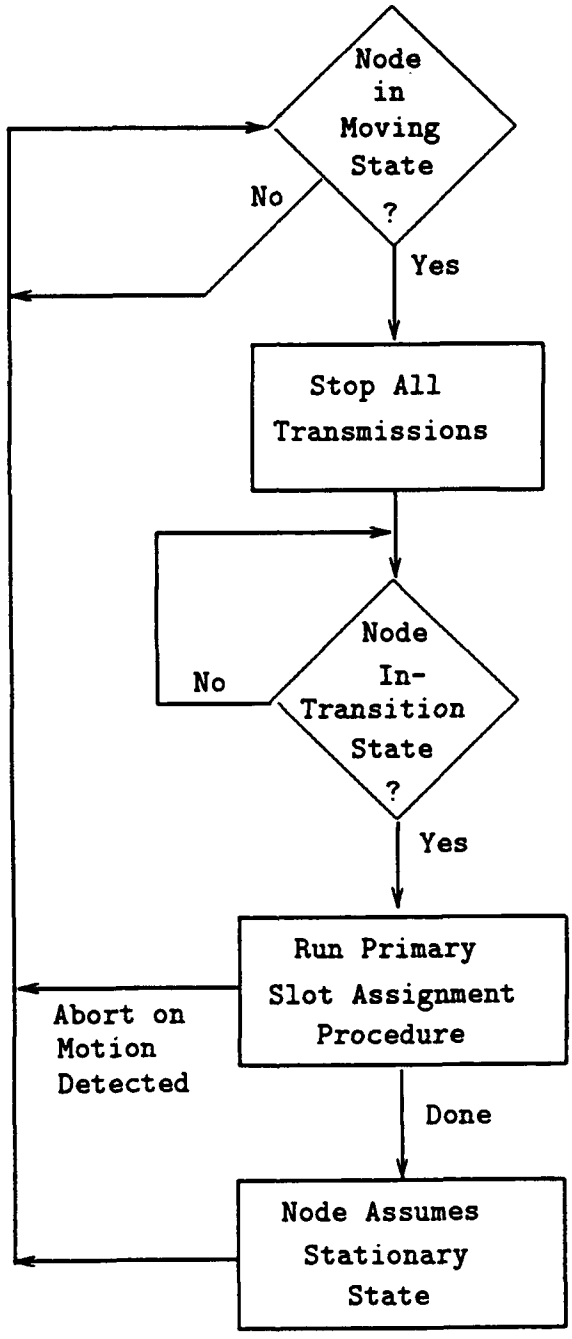

Figure 4: Primary_Slot_Assignment Procedure

schedules. The SSA and CR procedures are aborted if the conditions that allowed them to start (e.g., the node is Stationary) change while the procedures are running.

5 Primary_Slot_Assignment Procedure

The PSA procedure is divided into three steps. First a mobile node stops transmitting. Then to the best of its ability, it must learn the status of nodes in its new broadcast zone. Once this information is obtained it reassigns itself a new primary transmission slot. The node uses three sets to maintain its knowledge of its immediate neighbors. The first set, *Neighbor Packets (NP)*, contains all neighbors that the mobile node has learned about from the reception of status or data packets. The set *Status Packets (SP)* contains those neighbors from which the mobile node has received a status packet. The final set, *Collision Slots (CS)* contains those slots during which collisions are occurring. These sets are reset to empty once the node becomes Stationary and CS is set to empty after each NN packet it sends.

Figure 6 contains the pseudo-code for the PSA procedure. In Figure 7 we show the pseudo-code for the packet handler. This code describes the actions taken by nodes when they receive certain control packets.

Step 1: Stop Transmissions.
When a node moves it stops transmitting in all its allocated slots. This prevents its transmissions from interfering with any new neighbor's transmissions.

Step 2: Obtain a status packet from neighbors.
Once a node has completed moving it enters the In-Transition state and broadcasts a NEW_NEIGHBOR (NN) packet in the control slot. It then begins learning the status of its new broadcast zone. The NN packet contains normal status packet data and the contents of the two sets SP and CS, and a time stamp indicating the time of the first NN packet transmission.

Upon receipt of the NN packet, the receiving node will stop transmitting during any *secondary* transmission time slots that it is currently using. By examining the contents of the set SP, a neighbor can determine whether the mobile node knows its status. If it is in this set no further action is required at this point. If it does not find itself in set SP and its primary slot is not listed in set CS, then it will send a status packet during its next primary slot. If its primary slot is contained in the CS set, it stops transmissions during its primary slot. It delays a random period, say n frames, chosen uniformly between a minimum, n_{min}, and a maximum, n_{max}, value, and then transmits a status packet in the control slot. When a node stops transmitting during its primary slot, it must set a *restart primary timer* and reset it whenever a NN packet is received. This is necessary to prevent the node from remaining off the air indefinitely should the mobile node fail prior to the completion of the reassignment algorithm, or it fails to receive correctly the mobile node's

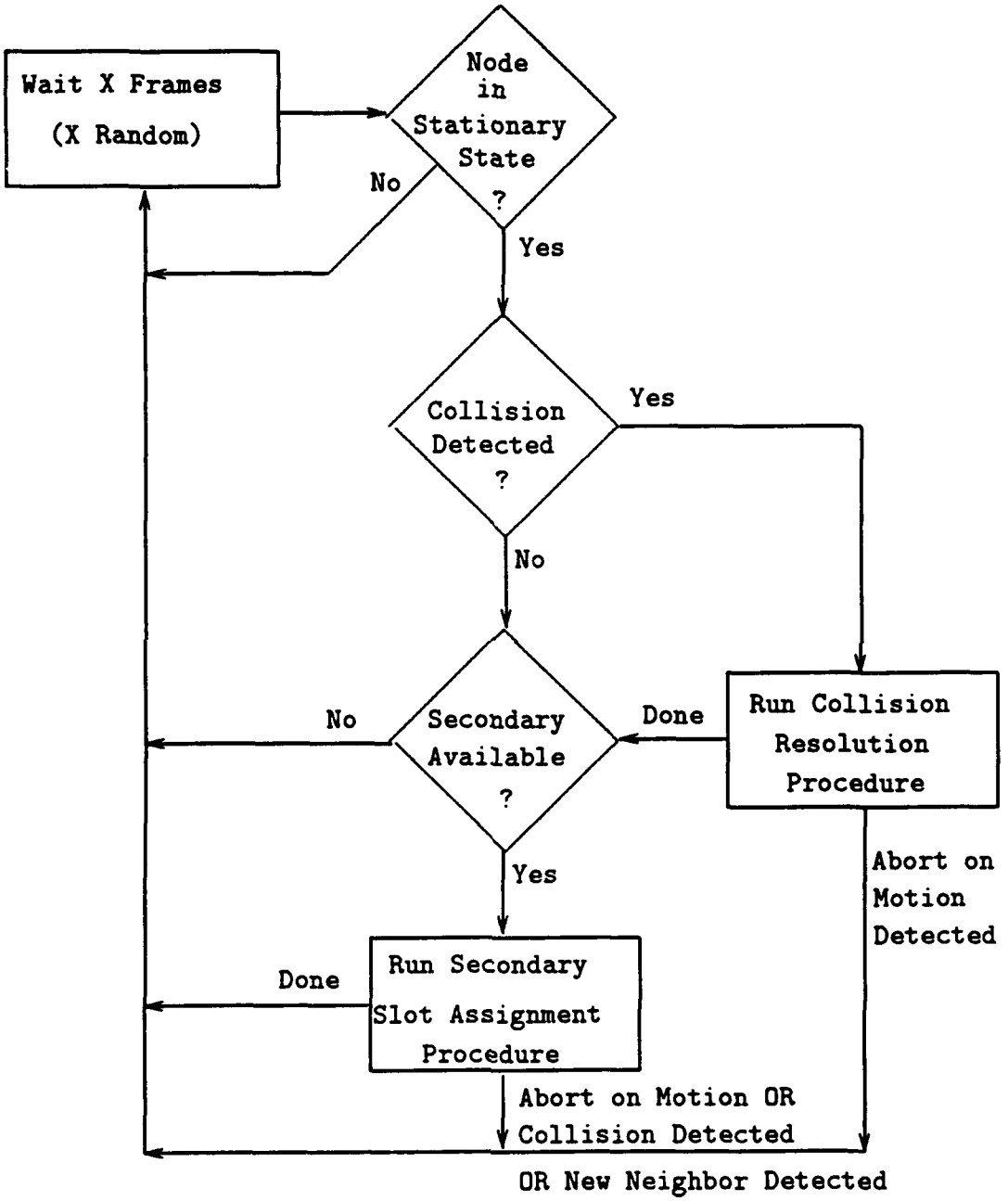

Figure 5: Secondary_Slot_Assignment and the Collision_Resolution Procedures

```
/* Data Structures */
    NP; /* set of known new neighbors */
    SP; /* set of neighbors having sent status packet */
    CS; /* set of slots during which collisions are heard */
procedure Primary_Slot_Assignment /* x ID of mobile node */
    Determine_Neighbors;
    if (Slot_Available) then
        select primary slot;
    else
        select slot in broadcast zone not used as primary slot;
    end if;
    send status packet on primary and control slots
end Primary_Slot_Assignment;

procedure Determine_Neighbors;
    constant
        max_tries := k; /* maximum number of tries for sending packets */
    variables
        tries; /* counter for number of packets sent */

    NP, SP, CS := φ;
    tries := 1;
    while (((NP ≠ SP) or (CS ≠ φ)) and (max_tries ≥ tries))
            or (SP = φ) do
        send NN packet;
        CS := φ;
        receive packets for two frames;
        if ((CS = φ) and (SP = φ) and (tries = 1)) or (previously delayed random peroid) then
            delay random period;
            tries := 0;
        end if;
        if (CS ≠ φ) and (tries > 1)then
            receive packets for n_max frames;
        end if;
        tries := tries + 1;
    end while;
end Determine_Neighbors;
```

Figure 6: Pseudo-code for Primary_Slot_Assignment Procedure

status packet when the mobile node does complete. If the timer expires, it resumes using its primary slot.

After sending the first NN packet, the mobile node receives packets during the next two frames. This is to insure that all neighbors have the opportunity to respond. (We assume that a node cannot immediately respond to a packet received in one slot in the next following slot.) During this waiting period, the mobile node will hear either data packets, status packets or collisions in response to its NN packet. It updates its status sets accordingly. After two frames, if CS and $SP = \phi$ then it will assume that its NN packet collided on the control channel and delay a random period of time prior to sending the next NN packet. If CS is not empty or $NP \neq SP$ it transmits another NN packet. If CS is empty it listens for another two frames. If CS is not empty it must wait some predetermined number of frames, n_{max}, to accommodate the random delay its neighbors will be waiting prior to responding with status packets using the control slot. Since nodes using the slots contained in CS stop transmitting in them, CS (except in the most unusual circumstances) will eventually become empty. This process continues until CS is empty and $NP = SP$, or these conditions are not met and the maximum allowable number of NN packets transmitted, *max_tries*, has been exceeded.

In the event that two mobile nodes are in the same area, and one hears a NN packet before it sends its own NN packet, the receiving node will abort its reassignment procedure. It sets a *restart PSA timer* and begins the procedure again when the timer expires or when it learns that the other mobile node has completed via status or data packets received from that node. If it has already sent a NN packet, then its first NN transmission time is compared with that of the time contained in the received NN packet. If its first transmission occurred after that of its new neighbor or if the times are equal and its ID is less than its neighbor then it aborts the reassignment procedure.

Step 3: Select new transmission slot.
When the mobile node has received a status packet from each of its neighbors, (or has exceeded the number of NN packets permitted), it will have a good (if not always complete) estimate of the status of its new broadcast zone. Since all neighbors have stopped transmitting during their secondary slots, the mobile node usually can find a free slot and select it as its new primary transmission slot. The boolean variable *Slot_Available* is True in that situation and can be computed from the information in the status packets.

The situation could arise where a mobile node is unable to select a primary transmission slot, even after all its neighbors have stopped transmitting during their secondary time slots, i.e., *Slot_Available* is False. Here, there will be at least one slot that is not available because one or more of its neighbors are receiving secondary transmissions during this slot from their neighbors. (Recall that the frame length is at least l^* which guarantees the availability of at least one slot per node.) In this event the node that moved selects one of these secondary slots as its primary.

To complete the algorithm, the mobile node sends an updated status packet on both its

new primary slot and on the control channel. Nodes receiving a status packet form a new neighbor, broadcasts a STOP_SECONDARY(*All*) (SS(*All*) packet to its neighbors. Nodes receiving a SS(*All*) packet give up their secondary slots. Thus all nodes within the mobile node's new broadcast zone will reselect secondary slots. This is done to insure fairness.

```
procedure Packet_Handler;
   /* Received packets indicate packet type, source and destination addresses*/
   Waiting_On := NIL;
   Primary_Hold := NIL;
   r := ID of receiving node;
   s := ID of packet sender;
   case packet type of
      Data:      if (State(r) = In-Transition) then
                    update set NP;
                    discard;
                 else if (State(r) = Stationary) then
                    handle normally;
                 end if;
      Collision: if (State(r) = In-Transition) then
                    update set CS;
                 else if (State(r) = Stationary) then
                    abort SSA procedure if running;
                    Collision_Detected := True;
                 end if;
      NN:        data
                    sets SP and CS;
                    s.time; /* Time 1st NN transmitted */
                 end data;
                 stop transmitting on secondary slots;
                 if (Secondary_Hold = φ) or (Secondary_Hold = s) then
                    set restart secondary timer;
                    Secondary_Hold := s;
                 end if;
                 if (Primary_Hold ∉ φ) then
                    reset restart primary timer;
                 end if;
                 if (Waiting_On ∉ φ) then
                    reset restart PSA timer;
                 end if;
                 if (State(r) = In-Transition) and (Waiting_On = φ) then
                    if ((r.time = s.time and (r < s)) or (r.time > s.time) then
                       abort PSA procedure;
                       Waiting_On := s;
                       set restart PSA timer;
                       /*On expiration of this timer the PSA procedure is restarted
                          and Waiting_On is set to NIL*/
                    end if;
                 end if;
```

```
            if (r's primary ∈ CS) and (Primary_Hold = NIL) then
                stop using primary slot;
                Primary_Hold := s;
                set restart primary timer;
                /* On expiration of this timer, r resumes using its primary
                    and Primary-Hold is set to NIL*/
            end if;
            if (r ∉SP) then
                if (primary slot stopped) then
                    delay random period;
                    send s a status packet on control channel;
                else
                    send s a status packet using primary slot;
                end if;
            end if;
Status:     data
                s's primary and secondary transmit slots;
                s's primary and secondary receive slots;
            end data;
            update view of broadcast zone;
            if (State(r) = In-Transition) then
                update sets NP, SP and CS;
                if (Waiting_On = s) then
                    start PSA procedure;
                    Waiting_On := NIL;
                    disable restart PSA timer;
                end if;
            end if;
            if (Primary_Hold = s) then
                resume transmissions using primary slot;
                Primary_Hold := NIL;
                disable restart primary timer;
            end if;
            if (Secondary_Hold = s) then
                number of secondary slots = 0;
                disable restart secondary timer;
                send SS(All secondary slots);
            end if;
```

```
    SDS:    data
                slot i;
            end data;
            send SS(i) to neighbor transmitting in slot i;
    SS:     data
                slot i;
            end data;
            stop transmitting in slot i;
            send status packet to neighbors;
    RP:     data
                s's primary and secondary transmit slots;
                s's primary and secondary receive slots;
            end data;
            update view of broadcast zone;
            stop transmitting on primary;
            if (r has secondary slot) then
                declare a secondary as new primary;
            else
                if (Secondary_Available) then
                    select an available slot as the primary slot;
                else
                    find usable slot i;
                    send SDS(i) or SS(i) to appropriate neighbor;
                    select (i) as primary slot;
                end if;
            end if;
            send status packet to neighbors
    end case;
End Packet_Handler;
```

Figure 7: Pseudo-code for Packet_Handler

6 Collision_Resolution Procedure

The reassignment algorithm makes every effort to ensure that a mobile node is aware of all nodes in its broadcast zone prior to allocating itself a new primary slot. Unfortunately, we cannot guarantee that this in fact will occur. It is possible that the PSA procedure runs to completion without the mobile node becoming aware of all its neighbors. Should this happen, collisions may eventually surface.

There are two types of collision that a node may experience. The first occurs when a node becomes aware that a neighbor is using one of its transmission slots to transmit in as well. This is a violation of condition C1 from Section 2 and we will refer to this kind of collision as a *Type 1* collision. This situation may happen, for example, when two mobile nodes relocate to a position such that they become neighbors, have no common neighbors, and transmit all their control packets simultaneously. Here neither will be aware of the other's existence because their packets will collide on the control channel. This can result in each selecting the same slot for their primary transmissions (and in the worst case any secondary slots as well). If each transmits all its packets simultaneously, they will never learn of each other. This is not in itself a serious problem because neither is affected by the collisions and the rest of the network operates as if the link between the two did not exist. Realistically though, one can assume that eventually one of these nodes will not transmit during its slot and will hear the other's transmission. At that point it will realize that it has a neighbor that is using its slot and collisions are happening.

The second type of collisions ensues when a node realizes that one or more of its neighbors are transmitting in the same slot. This type of collision violates condition C2. We will call this a *Type 2* collision. This can occur, for example, when two nodes x and z are using the same slot as their primary transmission slot and initially are not in each other's broadcast zone. Node y then moves to a position such that it can receive both x and z. This forces them into the same broadcast zone.

The node realizing that it is experiencing collisions must run the collision resolution algorithm. (See Figure 8.) The type of collision and the amount of information available to the node experiencing the collision, determines the actions that need to be taken to resolve the collision. For a Type 1 collision:

- If the node learning of the collision in slot i is using that slot for secondary transmissions, it stops transmission on it.
- If slot i is used as its primary slot then it can take one of several actions. First, if it has a secondary slot, it can select the secondary as its primary. If it does not have a secondary and knows the status of the offending neighbor, it can tell whether its neighbor is using the slot as a primary or secondary transmission slot.

```
procedure Collision_Resolution
    r := ID of node experiencing collision;
    i := collision slot;
    Z := set of neighbors transmitting in i who's status is known;
    case collision type of
        Type1:  if (i = r's secondary slot) then
                    stop transmitting during slot i;
                else if (i = r's primary) and
                    (r has secondary slot j) then
                    stop transmitting during slot i;
                    select slot j as new primary;
                else if (i = z's secondary,z ∈ Z) then
                    send SS(i) to z;
                else if (i = z's primary,z ∈ Z) and
                    (z has secondary slot j) then
                    send RP to z;
                else
                    run PSA procedure;
                end if;
        Type2:  if (Z = φ) then
                    Determine_Neighbors;
                else
                    send SS(i) to z ∈ Z that use i as secondary;
                    send RP to z ∈ Z using i as primary except one with highest ID;
                end if;
    end case;
end Collision_Resolution;
```

Figure 8: Collision Resolution Algorithm

- If it is a secondary slot, the node recognizing the collision can send a SS(i) packet over the control channel.
- If it is a collision that is occurring with a neighbor's primary slot and the neighbor has a secondary, it sends a REASSIGN_PRIMARY(RP) message to the neighbor over the control slot. A node receiving a RP packet (see Figure 7) will stop transmitting during its primary transmission slot and select a new one. This packet contains the sending node's status so that the receiving node has an updated view of its broadcast zone.

- If both nodes are using the slot as their primary slot and neither has a secondary, or the status of the neighbor is *not* known, the node experiencing the collision will execute the PSA procedure though it has not moved. This will make the other node(s) aware of its presence and allow it to reselect a primary slot that will not result in collisions.

When a Type 2 collision occurs in slot i:

- If the node experiencing the problem knows the status of the offending neighbors (i.e., it knows of at least one neighbor who is transmitting in this slot), it can determine if its neighbors are using the slot for secondary transmissions. To these neighbors it sends a SS(i). To all neighbors that are using slot i as their primary slot, the node would send RP packets in turn to each of those neighbors less the one with the highest ID (if it has identified more than one). This is done to insure that two or more nodes all sending RP packets to the same set of neighbors simultaneously send them to the same ordered set of nodes. Failing to do so could result in a never ending cycle of RP packets being transmitted. If the node receiving the RP packet does not have a secondary slot available, a new primary slot will only not be available because slots are being used by its neighbors for secondary transmissions or receptions. In this situation, the node sends a SS(i) or a SDS(i) packet to free a slot. It then selects this slot as its primary. Since the node is a Stationary node, it is unlikely to experience collisions caused by two or more of its neighbors primary transmissions occurring simultaneously. Thus the requirement to direct a neighbor to reselect a primary transmission slot, due to an inconsistent schedule caused by a mobile node, will not propagate throughout the network.

- If nothing is known about the offending nodes, the node experiencing the Type 2 collision executes the *Determine_Neighbor* routine of the PSA procedure to learn the identities of these nodes and resolve the collision.

Remarks

- Recall that the CR procedure is aborted if, while it is executing, the node starts moving to a new location. The reason is that the node is now entering a new neighborhood that may impose new requirements and collisions and thus any effort to resolve old collisions is wasted.
- As for the PSA procedure, there are no guarantees that the CR procedure will achieve its objective. Yet, if collisions persist, they will again be detected and the CR procedure will be run again.

7 Secondary_Slot_Assignment Procedure

A Stationary node that is not experiencing collisions and is not receipt of a neighbors NN packet, may attempt to allocate itself secondary slots. The availability of such slots is indicated by a boolean variable *Secondary_Available*. This variable is computed in each node based on the node's knowledge of the status of its broadcast zone. The SSA procedure allocates slots based on a fairness criterion γ_x, which we define as s_x/l_x, where s_x and l_x are the total number of slots assigned to node x, and the number of node x's neighbors respectively. We use this ratio as an estimator for the amount of traffic that a node will handle. If actual traffic rates are known, the arrival rate λ_x could easily be substituted for l_x. Yet in a dynamic environment this seems unlikely. (See [11] for a similar fairness criterion when traffic rates are assumed known.)

Each node computes a γ_x, which is disseminated, with the value of its *Secondary_Available* variable to the node's broadcast zone in normal periodic network status packets. Changes to either of these quantities initiate the sending of a status packet. A node upon receiving this information can determine if it should allocate itself an additional secondary slot. If a node has one or more slots available for secondary transmissions, and is not currently involved in the reassignment process nor detected a collision, it periodically executes the SSA procedure shown in Figure 9. A node assigns itself a secondary time slot if it is in possession of the smallest γ_x within its broadcast zone. It computes a new γ_x and then sends an updated status packet. If two or more nodes are competing for a secondary slot and have the same lowest γ_x value, then we use the node's ID as the discriminator.

Remarks

- The SSA procedure is aborted if a collision or movement is detected. This is because any secondary slot assignments made are rendered useless by the node's movement and is suspicious if collisions are detected.

- Note that if several secondary slots are available to a node, only one is assigned per execution of the SSA procedure. This will allow a node's broadcast zone time to react to the assignment before another one is made at the same node. Consecutive executions are separated by X frames (see Figure 5).

- As in the CR and PSA procedures no guarantees are made that the secondary slot selection will provide for a consistent TDMA schedule. Also as before the mechanisms already described should ultimately resolve any inconsistencies.

  - An example of the type of problem that can occur is when two nodes in the same broadcast zone select the same secondary slot when a node they have as a common neighbor relocates and makes additional slots available. For instance, suppose two nodes x and y have two neighbors in common, nodes a and b, but they themselves are not directly linked. Now assume that node b departs the zone, freeing a single slot that both nodes x and y could use. Each will believe that it can allocate the freed slot to itself because its current view of the broadcast zone suggests that it is the only node with an available slot. Both would begin using the slot and collisions would result at node a. The requirement that the SSA procedure be run periodically at random intervals will reduce the possibility of this situation.
  - Another problem that can arise is when a node gives up its secondary slots as a result of receiving a NN packet (see Figure 7) from a node that has just relocated. These slots might be picked up by one of the node's Stationary neighbors before it is made available to the mobile node. The likelihood of this is reduced since a node does not instantaneously react to the availability of a secondary slot.

8 Examples

We will now demonstrate how the procedures would function using our previous sample network shown in Figure 2 with the initial primary and secondary assignments given in Table 1. In this example node H relocates and establishes a new link with node F as in Figure 3.

The first step requires that node H stop transmitting. Once it has relocated it transmits a NN packet on the control channel that is received by nodes C, F and G. Only node C has a secondary slot. It will stop using it and compute a new γ_c. Each will transmit a status packet to node H during their primary slot. In this example, collisions will not occur and after two frames node H will have received the status packets from each of its new neighbors. At this point, the sets $NP = SP$ and $CS = \phi$. Node H will assume that it has

```
procedure Secondary_Slot_Assignment;
    r := ID of node running procedure;
    Y := Set of nodes in r's broadcast zone with available secondary slots;
    Y_min := {x ∈ Y | γ_x = min_{y∈Y} γ_y};
        if ( γ_r < γ_y for all y ∈ Y) then
                choose secondary slot;
                else if (γ_r = min_{y∈Y} γ_y) and (r < x for all x ∈ Y_min) then
                    choose secondary slot;
                end if;
        end if;
end Secondary_Slot_Assignment;
```

Figure 9: Secondary_Slot_Assignment Procedure

learned the status of its new broadcast zone and select slot 8, freed by node C, as its new primary slot. The algorithm completes with node H broadcasting its new status.

Assume for a moment that node H, in response to its NN packet, did not receive a status packet from node C nor any of C's data packets. Then it would assume that its new neighbors were only nodes F and G. After the two frames, it would incorrectly surmise that slot 1 was the first available free slot. Node C would soon begin detecting Type 2 collisions during slot 1. If we assume that it had correctly received H's status packet, then when it executes the CR procedure, it will send a RP to node D. Node D would stop using slot 1 as its primary and select slot 5, its only secondary as its new primary. If on the other hand, node C did not know either node D or H's ID nor their status, node C in this situation must run the Determine_Neighbors procedure to discover the identity of nodes D and H. Note that set CS now contains slot 1. Both node D and H are required by the Determine_Neighbors procedure to stop transmissions during this slot, delay a random period and then send their status packets using the control slot. This continues until node C sends a NN packet with their ID in the set SP or their restart primary timer expires. When node C has correctly learned their status, it will send a RP to node D. If we again presume the worse by assuming that node D fails to receive the RP packet, it will continue using slot 1 and Type 2 collisions will again be experienced at node C. However this time, node C will know the status of the two offending nodes and simply transmit a RP packets to node D until the collisions are resolved.

We now explore the situation when two nodes move into the same neighborhood. In this example we let nodes E and H relocate to a position such that they can now receive only nodes A and D, and each other (see Figure 10). First we will assume that collisions and errors do not take place and then walk through the sequence of events when collisions and errors do occur.

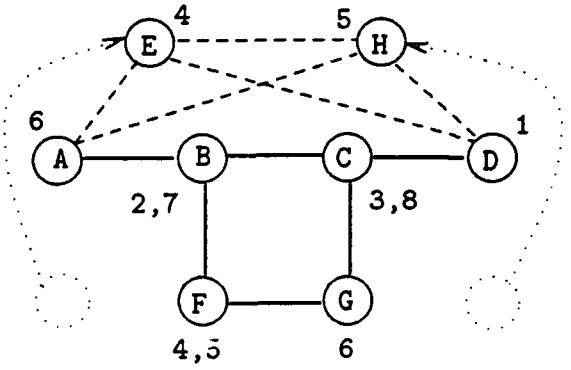

Figure 10: Reassignment Example - Final Assignments

Both nodes will stop transmitting in their primary slot. We will assume that node E's NN packet is transmitted first without error. Node H receives E's NN packet, aborts the PSA procedure and sets its restart PSA timer. When this packet is received by nodes A and D, they will stop transmitting during their secondary slots 6 and 5 respectively. Each will then transmit a status packet during slot 1. Node E will hear the two status packet collide. After two frames $NP = SP = \phi$ and $CS = \{1\}$ and a second NN packet will be transmitted reflecting these facts. This will cause node H to reset it restart PSA timer. Nodes A and D will see that they are not in the set SP and that their primary slot is in the set CS. They will each stop transmitting during their primary slot, delay a random period and send a status packet on the control channel. After Node E receives their status packets, $NP = SP = \{A, D\}$ and $CS = \phi$. Node E selects slot 4 as its primary transmission slot. It then sends an updated status packet on slot 4 and on the control channel. Node D, not hearing any collisions, will run the SSA procedure and reselect slot 5 as its secondary slot. Node H, upon hearing E's status packet on the control channel, restarts the PSA procedure. Its execution of the PSA algorithm will result in node D again relinquishing slot 5 and node H selecting it as its primary. Over time nodes F and G will realize that slot 4 is available. Execution of the SSA routine will result in node F selecting slot 4 as a secondary slot. Node E will also eventually detect that both nodes A and D are using slot 1 as their primary slot and run the CR procedure that results in Node E sending a RP to node A. Node A discards slot 1 as its primary and selects slot 6 as its new primary.

Now we examine the same example but do not assume an ideal sequence of events. First suppose that both nodes E and H transmit their NN packets simultaneously and that they collide at all neighbors. Since neither will receive a response, both will assume a collision has occurred, delay a random period and send another NN packet. If we again assume the worst, both will continue the process unaware of each other . This will result in each selecting slot 4 as their primary slot. Eventually one would detect a Type 1 collision or

nodes A or D would experience a Type 2 collision. The node detecting the collision will attempt to resolve it be running the CR procedure.

In another sequence suppose that neither nodes E nor H stops for the other, node E learns only of A's existence and node H learns only of D's presence. In this case neither node E nor H will recognize the potential for collision during slot 1 and may choose it as their primary slot. Again, after they complete the algorithm, both Type 1 and 2 collisions will be occurring and the nodes experiencing the collision will take the appropriate action.

9 Performance Of the Reassignment Procedures

The procedures presented above will function correctly and efficiently if errors do not occur and only one node is permitted to move at a time. Unfortunately these assumptions are not realistic. Therefore, our solution is complicated by the fact that we are required to incorporate additional features to handle operations under adverse conditions.

There are many parameters that will affect the operational performance of the procedures that we have proposed. First we assume that when a node moves the time required for the schedule to "settle" is short compared to the length of time between consecutive moves. If the settling time is longer than assumed or moves occur with great frequency than anticipated, the mobile node may not be able to communicate with other network nodes for a significant amount of time. Also if all nodes are constantly moving, the network may not be capable of functioning at all. Two other factors that will have a direct effect on the settling time are the frame length and the sampling interval (defined in Figure 5 as the random variable X) before starting the collision resolution or secondary slot assignment procedure. The longer the sampling interval the greater time it will take to reach a consistent schedule. Short sampling intervals, however, will increase the likelihood of concurrent or close to concurrent execution of the SSA and CR procedures. A short frame length (recall it is always $\geq L^*$ given in (1)) will provide nodes with less secondary slots and make schedule changes more time consuming. A longer frame length, on the other hand, although providing nodes with more secondary slots, may adversely affect network delays. The extent of this effect is unclear and will need to be examined.

The reliability of the channels is another factor that will affect the efficiency of these procedures. While we assume that errors may occur, if the error rate is too great, too many control and status packets will not reach their intended destinations. This can result in inconsistent schedules, which because of the high rate of transmission errors, may never become consistent.

We are in the process of developing a simulation which implements these procedures. Besides testing and measuring the above factors that will influence the operational feasibility of our methodology, we hope to evaluate other performance measures as well. Some of these include the relative time that it takes a mobile node to reestablish a collision free schedule

and how long it is unable to communicate. We want to also measure how often other nodes are affected and how much time a mobile node's neighbors are forced to temporarily forego transmissions on their primary slot.

10 Conclusion

We considered the problem of locally reassigning time slots to nodes in a mobile packet radio network and simultaneously maximizing the use of the bandwidth. We first examined what appears to be conflicting goals: maximal use of the bandwidth and local rescheduling of a TDMA frame. We then presented three procedures to resolve the problem. The first was a Primary_Slot_Assignment procedure that allows mobile nodes to make their best effort at learning of the slots in use for their broadcast zone and then finding an available primary slot during which it can transmit without collision. Because we do not assume the existence of a reliable collision free control channel for the exchange of network control packets, we cannot guarantee the resulting TDMA schedule will be collision free. Thus when collisions are detected, a Collision_Resolution procedure must be executed to restore the schedule to one that does not contain potential for collisions. Finally we presented a Secondary_Slot_Assignment procedure that allocates any free slots, based on a fairness criteria, as secondary transmission slots. The procedure requires knowledge of only the status and transmission schedules of the nodes in the broadcast zone. The resulting schedule is one that uses all available bandwidth.

These three procedures insure that a mobile node does not involve the entire network in the reassignment process. Initially only a mobile node's immediate neighbors are involved. They are required to stop transmitting during their secondary slots and on occasion reselect a primary slot. Neighbors two hops away become involved if the mobile node cannot find a new primary slot or collisions occur between the mobile node's immediate neighbors. Here a two hop away neighbor might be required to give up one of their secondary slots.

References

[1] Randolph Nelson and Leonard Kleinrock. "Spatial TDMA: A collision-free mulithop channel access protocol". *IEEE Transactions on Communications*, COM-33(9):934–944, September 1985.

[2] Thuan V. Truong. "TDMA in mobile radio network: An assessment of certain approaches". In *IEEE GLOBECOM 84*, pages 504–507, 1984.

[3] Israel Cidon and Moshe Sidi. "Distributed assignment algorithms for multihop packet radio networks". *IEEE Transactions on Computers*, 38(10):1353–1361, October 1989.

[4] Roger Merk Clifford Warner and et al. "Handoff assigned multiple access (HAMA): A new multiple access protocol for computer communicaion networks". In *IEEE MILCOM 83*, pages 183–189, 1983.

[5] Imrich Chlamtac and Shay Kutten. "On broadcasting in radio networks - problem analysis and protocol design". *IEEE Transactions on Communications*, COM-33(12):1240–1246, December 1985.

[6] Imrich Chlamtac and Yishay Mansour. "Local cycle generation in multihop radio networks". In *IEE MILCOM 87*, pages 801–806, 1987.

[7] Zvi Rosberg and Moshe Sidi. "TDM policies in multistation packet radio networks". *IEEE Transactions on Communications*, COM-37(1):31–38, January 1989.

[8] Imrich Chlamtac and Sholmit S. Pinter. "Distributed nodes organization algorithm for channel access in a multihop dynamic radio network". *IEEE Transactions on Computers*, C-36(6):728–737, June 1987.

[9] Imrich Chlamtac and A. Lerner. "A link allocation protocol for mobile multihop radio networks". In *IEEE MILCOM 85*, pages 238–242, 1985.

[10] P.E. Sarachik M.J. Post and A.S. Kershenbaum. "A biased greedy algorithm for scheduling multi-hop radio networks". In *Procedings Information Sciences and Systems*, pages 564–572, March 1985.

[11] A. Ephremides and T.V. Truong. "Scheduling broadcasts in multihop radio networks". *IEEE Transactions on Communications*, COMM-38(4):456–460, April 1990.

[12] Imrich Chlamtac and Anat Lerner. "Fair algorithms for maximal link activation in multihop radio networks". *IEEE Transactions on Communications*, COM-35(7):739–746, July 1987.

[13] David S. Stevens and Mostafa H. Ammar. "Evaluation of slot allocation strategies for TDMA protocols in packet radio networks". In *MILCOM 90*, pages 835–839, September 1990.

[14] Imrich Chlamtac and Shay Kutten. "Tree-based broadcasting in multihop radio networks". *IEEE Transactions on Computers*, 36(10):1209–1223, October 1987.